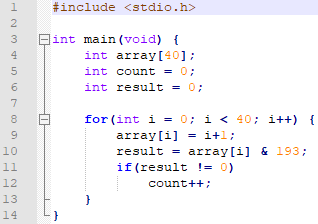
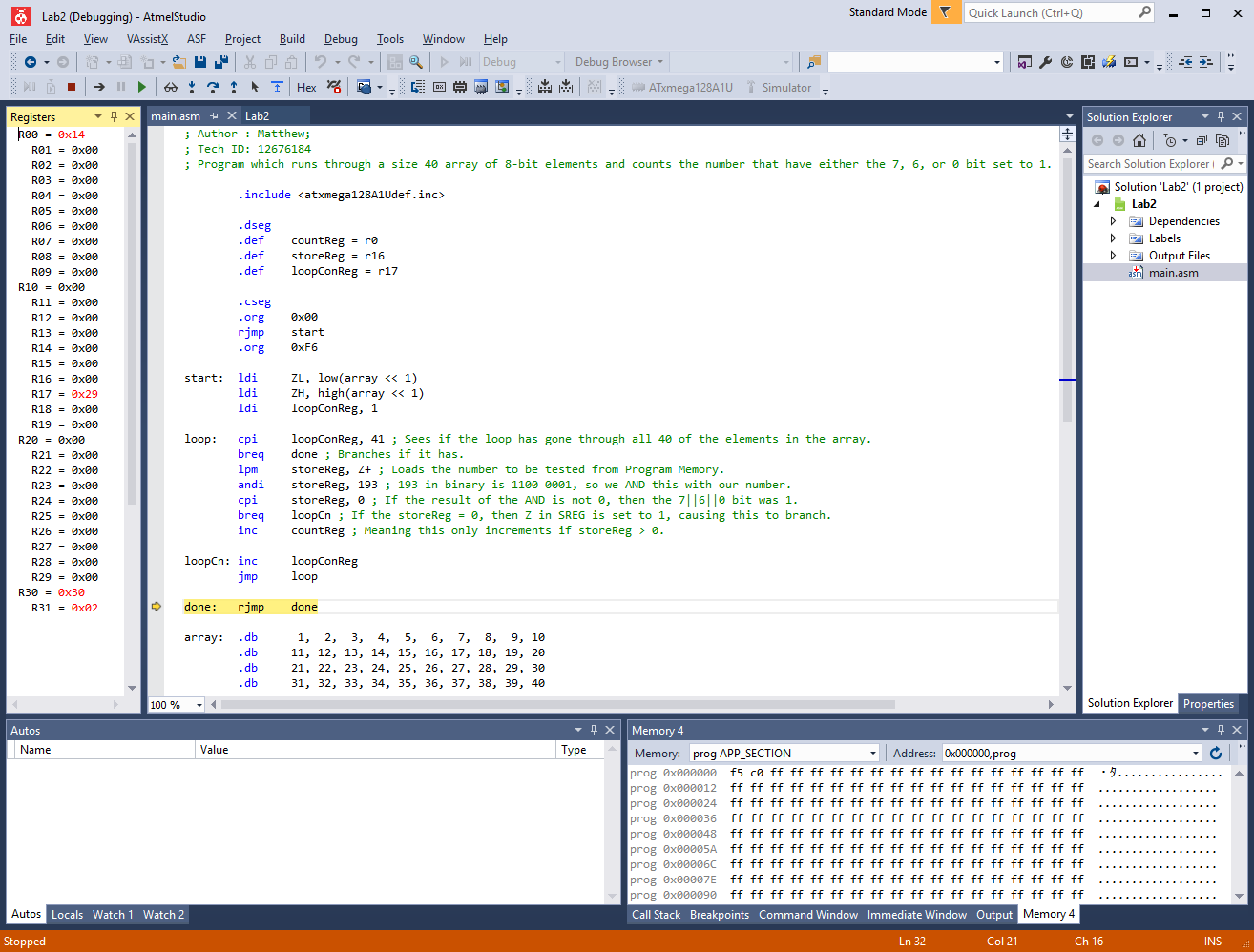
Here is an equivalent program in C:



We want to do a bitwise AND operation between the number we’re testing and the number 193, since that number’s binary representation is 1100 0001. Thus, all the bits we want to test for are set in this number, meaning that when we AND them together the result of the AND will be 0 if the 7th, 6th, and 0th bits of the number being tested were not set, and some number bigger than 0 (specifically it could be 193, 192, 129, 128, 65, 64, or 1) if some combination of those three bits are set.

As for evidence that the program is working, here is a screenshot:



Since the numbers are 1 through 40, none of them are above 64 or 128, so the 6th and 7th bits will never be set. Thus, the only bit we care about is the 0th, which will be set if a number is odd. Since we have 20 odd numbers in our database, our result should be 20. If we look at our countReg (r0), we can see it has the value of , meaning our program worked as intended.